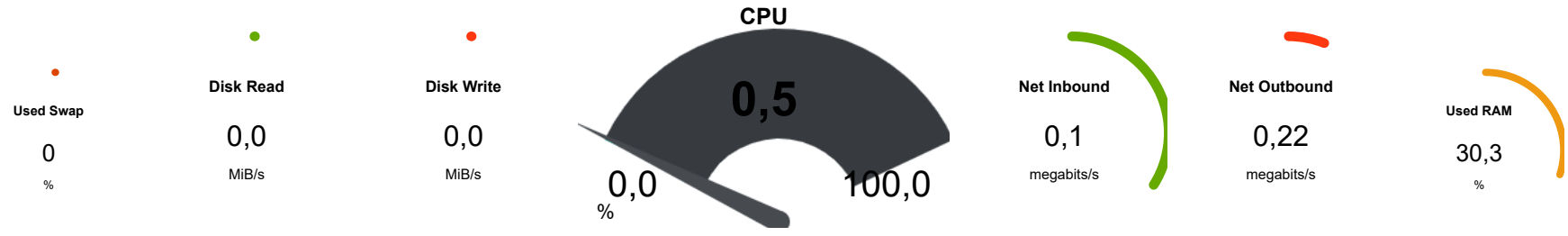
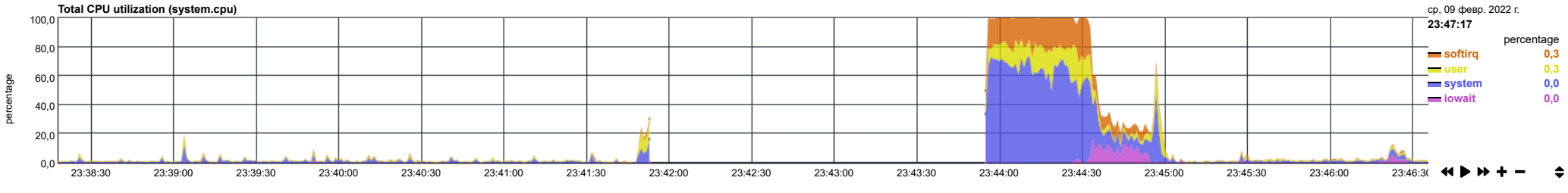
System Overview

Overview of the key system metrics.



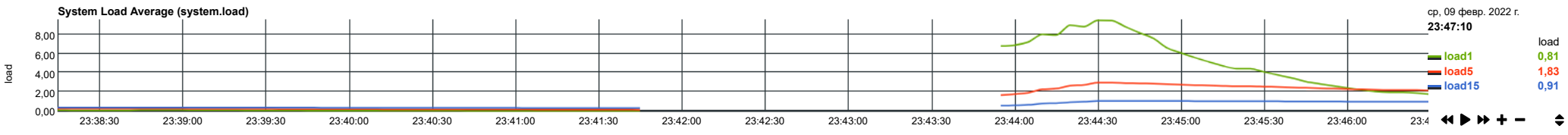
cpu

Total CPU utilization (all cores). 100% here means there is no CPU idle time at all. You can get per core usage at the CPUs section and per application usage at the Applications Monitoring section.
Keep an eye on **iowait** (0,00%). If it is constantly high, your disks are a bottleneck and they slow your system down.
An important metric worth monitoring, is **softirq** (0,26%). A constantly high percentage of softirq may indicate network driver issues.



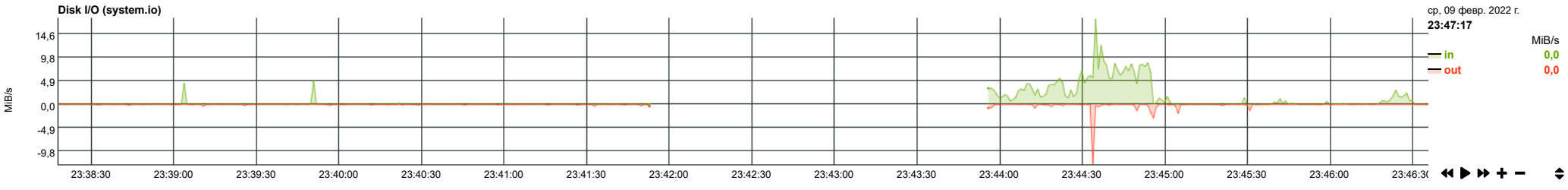
load

Current system load, i.e. the number of processes using CPU or waiting for system resources (usually CPU and disk). The 3 metrics refer to 1, 5 and 15 minute averages. The system calculates this once every 5 seconds. For more information check this wikipedia article (https://en.wikipedia.org/wiki/Load_(computing))



disk

Total Disk I/O, for all physical disks. You can get detailed information about each disk at the Disks section and per application Disk usage at the Applications Monitoring section. Physical are all the disks that are listed in `/sys/block` , but do not exist in `/sys/devices/virtual/block` .



Memory paged from/to disk. This is usually the total disk I/O of the system.

netdata

ram

System Random Access Memory (i.e. physical memory) usage.

swap

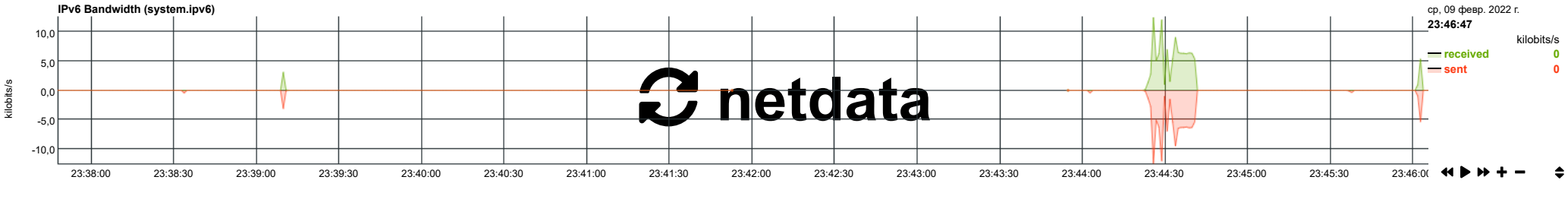
System swap memory usage. Swap space is used when the amount of physical memory (RAM) is full. When the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space (usually a disk, a disk partition or a file).

network

Total bandwidth of all physical network interfaces. This does not include `lo` , VPNs, network bridges, IFB devices, bond interfaces, etc. Only the bandwidth of physical network interfaces is aggregated. Physical are all the network interfaces that are listed in `/proc/net/dev` , but do not exist in `/sys/devices/virtual/net` .

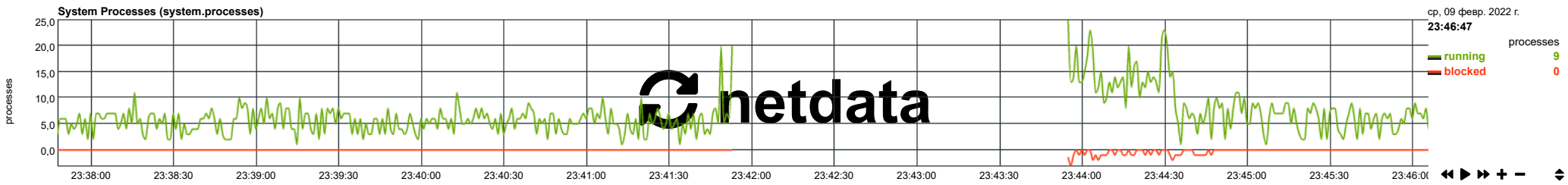
Total IP traffic in the system.

Total IPv6 Traffic.

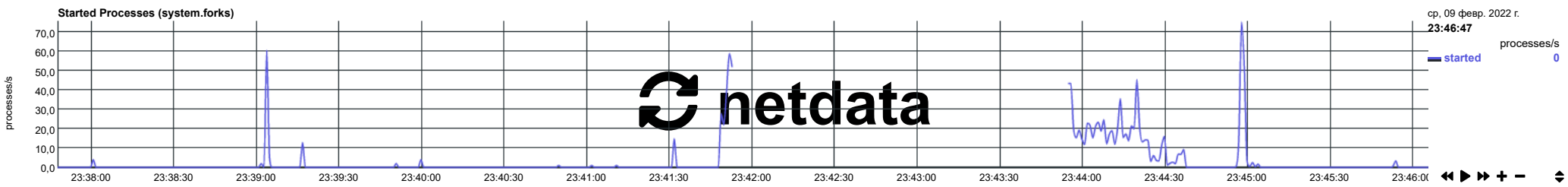


processes

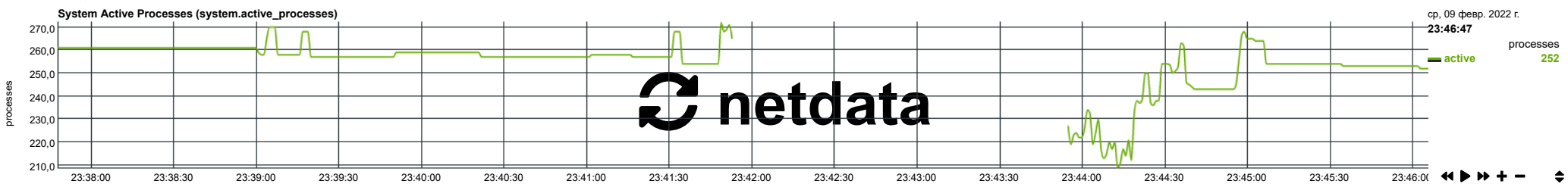
System processes. **Running** are the processes in the CPU. **Blocked** are processes that are willing to enter the CPU, but they cannot, e.g. because they wait for disk activity.



Number of new processes created.



All system processes.



Context Switches (https://en.wikipedia.org/wiki/Context_switch), is the switching of the CPU from one process, task or thread to another. If there are many processes or threads willing to execute and very few CPU cores available to handle them, the system is making more context switching to balance the CPU resources among them. The whole process is computationally intensive. The more the context switches, the slower the system gets.

idlejitter

Idle jitter is calculated by netdata. A thread is spawned that requests to sleep for a few microseconds. When the system wakes it up, it measures how many microseconds have passed. The difference between the requested and the actual duration of the sleep, is the **idle jitter**. This number is useful in real-time environments, where CPU jitter can affect the quality of the service (like VoIP media gateways).

interrupts

Total number of CPU interrupts. Check `system.interrupts` that gives more detail about each interrupt and also the CPUs section where interrupts are analyzed per CPU core.

CPU interrupts in detail. At the CPUs section, interrupts are analyzed per CPU core.

softirqs

CPU softirqs in detail. At the CPUs section, softirqs are analyzed per CPU core.

softnet

Statistics for CPUs SoftIRQs related to network receive work. Break down per CPU core can be found at CPU / softnet statistics. **processed** states the number of packets processed, **dropped** is the number packets dropped because the network device backlog was full (to fix them on Linux use `sysctl` to increase `net.core.netdev_max_backlog`), **squeezed** is the number of packets dropped because the network device budget ran out (to fix them on Linux use `sysctl` to increase `net.core.netdev_budget` and/or `net.core.netdev_budget_usecs`). More information about identifying and troubleshooting network driver related issues can be found at Red Hat Enterprise Linux Network Performance Tuning Guide (https://access.redhat.com/sites/default/files/attachments/20150325_network_performance_tuning.pdf).

entropy

Entropy (https://en.wikipedia.org/wiki/Entropy_(computing)), is a pool of random numbers (/dev/random (https://en.wikipedia.org/wiki//dev/random)) that is mainly used in cryptography. If the pool of entropy gets empty, processes requiring random numbers may run a lot slower (it depends on the interface each program uses), waiting for the pool to be replenished. Ideally a system with high entropy demands should have a hardware device for that purpose (TPM is one such device). There are also several software-only options you may install, like `haveged`, although these are generally useful only in servers.

uptime

ipc semaphores

ipc shared memory

CPUs

Detailed information for each CPU of the system. A summary of the system for all CPUs can be found at the System Overview section.

utilization

interrupts

softirqs

softnet

Statistics for per CPUs core SoftIRQs related to network receive work. Total for all CPU cores can be found at System / softnet statistics. **processed** states the number of packets processed, **dropped** is the number packets dropped because the network device backlog was full (to fix them on Linux use `sysctl` to increase `net.core.netdev_max_backlog`), **squeezed** is the number of packets dropped because the network device budget ran out (to fix them on Linux use `sysctl` to increase `net.core.netdev_budget` and/or `net.core.netdev_budget_usecs`). More information about identifying and troubleshooting network driver related issues can be found at Red Hat Enterprise Linux Network Performance Tuning Guide (https://access.redhat.com/sites/default/files/attachments/20150325_network_performance_tuning.pdf).

cpuidle

Memory

Detailed information about the memory management of the system.

system

Available Memory is estimated by the kernel, as the amount of RAM that can be used by userspace processes, without causing swapping.

Committed Memory, is the sum of all memory which has been allocated by processes.

A page fault (https://en.wikipedia.org/wiki/Page_fault) is a type of interrupt, called trap, raised by computer hardware when a running program accesses a memory page that is mapped into the virtual address space, but not actually loaded into main memory. If the page is loaded in memory at the time the fault is generated, but is not marked in the memory management unit as being loaded in memory, then it is called a **minor** or soft page fault. A **major** page fault is generated when the system needs to load the memory page from disk or swap memory.

kernel

Dirty is the amount of memory waiting to be written to disk. **Writeback** is how much memory is actively being written to disk.

The total amount of memory being used by the kernel. **Slab** is the amount of memory used by the kernel to cache data structures for its own use. **KernelStack** is the amount of memory allocated for each task done by the kernel. **PageTables** is the amount of memory decicated to the lowest level of page tables (A page table is used to turn a virtual address into a physical memory address). **VmallocUsed** is the amount of memory being used as virtual address space.

slab

Reclaimable is the amount of memory which the kernel can reuse. **Unreclaimable** can not be reused even when the kernel is lacking memory.

Disks

Charts with performance information for all the system disks. Special care has been given to present disk performance metrics in a way compatible with `iostat -x`. netdata by default prevents rendering performance charts for individual partitions and unmounted virtual disks. Disabled charts can still be enabled by configuring the relative settings in the netdata configuration file.

/

Amount of data transferred to and from disk.

Completed disk I/O operations. Keep in mind the number of operations requested might be higher, since the system is able to merge adjacent to each other (see merged operations chart).

I/O operations currently in progress. This metric is a snapshot - it is not an average over the last interval.

Backlog is an indication of the duration of pending disk operations. On every I/O event the system is multiplying the time spent doing I/O since the last update of this field with the number of pending operations. While not accurate, this metric can provide an indication of the expected completion time of the operations in progress.

Disk Utilization measures the amount of time the disk was busy with something. This is not related to its performance. 100% means that the system always had an outstanding operation on the disk. Keep in mind that depending on the underlying technology of the disk, 100% here may or may not be an indication of congestion.

The average time for I/O requests issued to the device to be served. This includes the time spent by the requests in queue and the time spent servicing them.

The average I/O operation size.

The average service time for completed I/O operations. This metric is calculated using the total busy time of the disk and the number of completed operations. If the disk is able to execute multiple parallel operations the reporting average service time will be misleading.

The sum of the duration of all completed I/O operations. This number can exceed the interval if the disk is able to execute I/O operations in parallel.

sda

Amount of data transferred to and from disk.

Completed disk I/O operations. Keep in mind the number of operations requested might be higher, since the system is able to merge adjacent to each other (see merged operations chart).

Backlog is an indication of the duration of pending disk operations. On every I/O event the system is multiplying the time spent doing I/O since the last update of this field with the number of pending operations. While not accurate, this metric can provide an indication of the expected completion time of the operations in progress.

Disk Utilization measures the amount of time the disk was busy with something. This is not related to its performance. 100% means that the system always had an outstanding operation on the disk. Keep in mind that depending on the underlying technology of the disk, 100% here may or may not be an indication of congestion.

The average time for I/O requests issued to the device to be served. This includes the time spent by the requests in queue and the time spent servicing them.

The average I/O operation size.

The average service time for completed I/O operations. This metric is calculated using the total busy time of the disk and the number of completed operations. If the disk is able to execute multiple parallel operations the reporting average service time will be misleading.

The number of merged disk operations. The system is able to merge adjacent I/O operations, for example two 4KB reads can become one 8KB read before given to disk.

The sum of the duration of all completed I/O operations. This number can exceed the interval if the disk is able to execute I/O operations in parallel.

/dev

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/dev/shm

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/run

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/run/lock

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/run/netdata

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/run/snapd/ns

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/tmp

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/vagrant

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/var/cache/netdata

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/var/lib/netdata

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/var/log

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

/var/tmp

Disk space utilization. reserved for root is automatically reserved by the system to prevent the root user from getting out of space.

inodes (or index nodes) are filesystem objects (e.g. files and directories). On many types of file system implementations, the maximum number of inodes is fixed at filesystem creation, limiting the maximum number of files the filesystem can hold. It is possible for a device to run out of inodes. When this happens, new files cannot be created on the device, even though there may be free space available.

Networking Stack

Metrics for the networking stack of the system. These metrics are collected from `/proc/net/netstat`, apply to both IPv4 and IPv6 traffic and are related to operation of the kernel networking stack.

tcp

TCP connection aborts. **baddata** (`TCPAbortOnData`) happens while the connection is on `FIN_WAIT1` and the kernel receives a packet with a sequence number beyond the last one for this connection - the kernel responds with `RST` (closes the connection). **userclosed** (`TCPAbortOnClose`) happens when the kernel receives data on an already closed connection and responds with `RST` . **nomemory** (`TCPAbortOnMemory` happens when there are too many orphaned sockets (not attached to an fd) and the kernel has to drop a connection - sometimes it will send an `RST` , sometimes it won't. **timeout** (`TCPAbortOnTimeout`) happens when a connection times out. **linger** (`TCPAbortOnLinger`) happens when the kernel killed a socket that was already closed by the a...
show more information

ecn

Explicit Congestion Notification (ECN) (https://en.wikipedia.org/wiki/Explicit_Congestion_Notification) is a TCP extension that allows end-to-end notification of network congestion without dropping packets. ECN is an optional feature that may be used between two ECN-enabled endpoints when the underlying network infrastructure also supports it.

IPv4 Networking

Metrics for the IPv4 stack of the system. Internet Protocol version 4 (IPv4) (https://en.wikipedia.org/wiki/IPv4) is the fourth version of the Internet Protocol (IP). It is one of the core protocols of standards-based internetworking methods in the Internet. IPv4 is a connectionless protocol for use on packet-switched networks. It operates on a best effort delivery model, in that it does not guarantee delivery, nor does it assure proper sequencing or avoidance of duplicate delivery. These aspects, including data integrity, are addressed by an upper layer transport protocol, such as the Transmission Control Protocol (TCP).

sockets

packets

errors

icmp

tcp

The number of established TCP connections (known as `CurrEstab`). This is a snapshot of the established connections at the time of measurement (i.e. a connection established and a connection disconnected within the same iteration will not affect this metric).

active or `ActiveOpens` is the number of outgoing TCP **connections attempted** by this host. **passive** or `PassiveOpens` is the number of incoming TCP **connections accepted** by this host.

`EstabResets` is the number of established connections resets (i.e. connections that made a direct transition from `ESTABLISHED` or `CLOSE_WAIT` to `CLOSED`). `OutRsts` is the number of TCP segments sent, with the `RST` flag set (for both IPv4 and IPv6). `AttemptFails` is the number of times TCP connections made a direct transition from either `SYN_SENT` or `SYN_RECV` to `CLOSED`, plus the number of times TCP connections made a direct transition from the `SYN_RECV` to `LISTEN`. `TCPSynRetrans` shows retries for new outbound TCP connections, which can indicate general connectivity issues or backlog on the remote host.

udp

IPv6 Networking

Metrics for the IPv6 stack of the system. Internet Protocol version 6 (IPv6) (https://en.wikipedia.org/wiki/IPv6) is the most recent version of the Internet Protocol (IP), the communications protocol that provides an identification and location system for computers on networks and routes traffic across the Internet. IPv6 was developed by the Internet Engineering Task Force (IETF) to deal with the long-anticipated problem of IPv4 address exhaustion. IPv6 is intended to replace IPv4.

packets

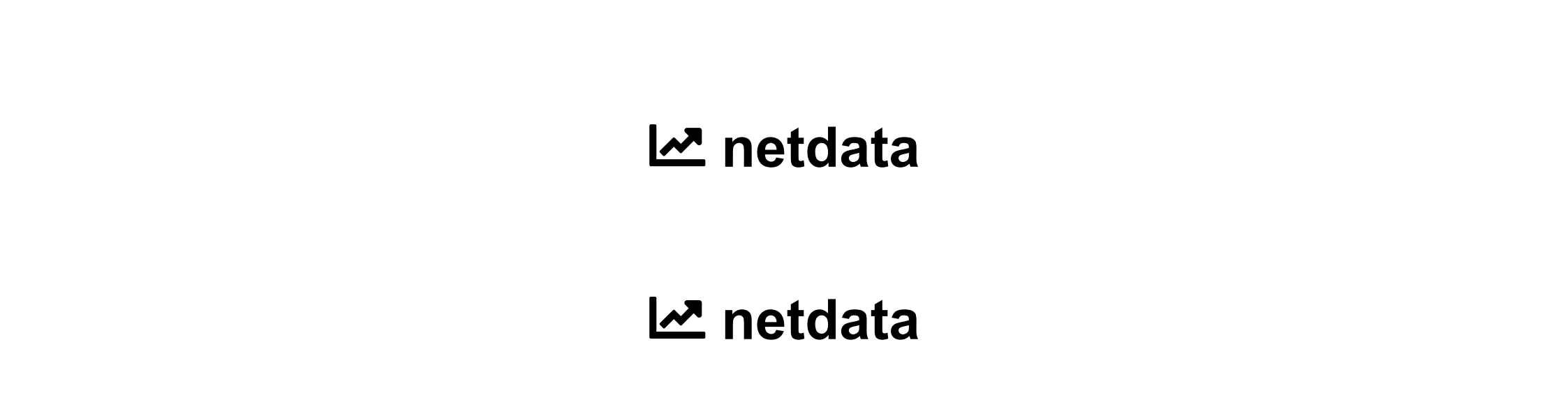
tcp6

udp6

raw6

multicast6

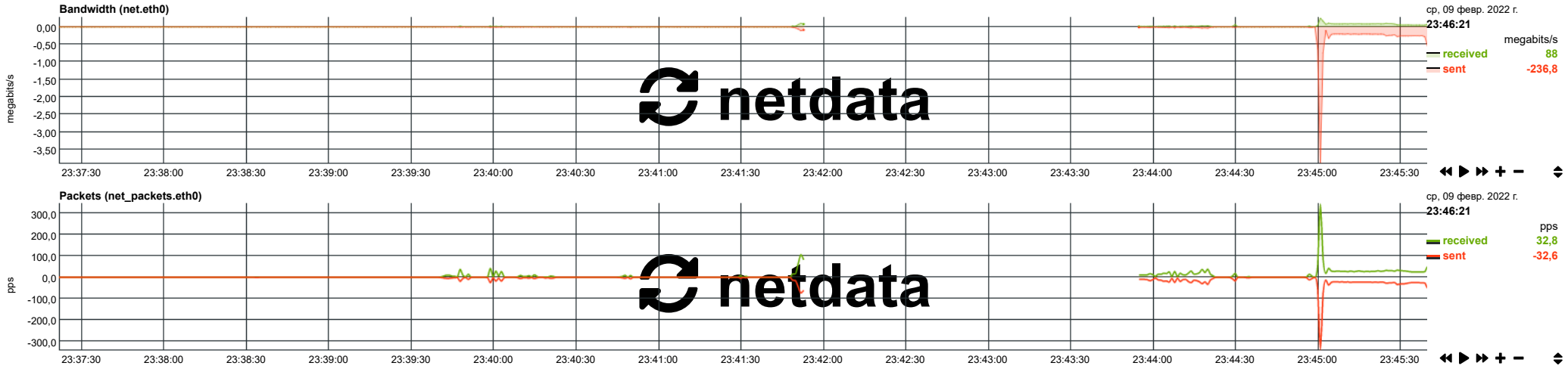
icmp6



Network Interfaces

Performance metrics for network interfaces.

eth0



Power Supply

Statistics for the various system power supplies. Data collected from Linux power supply class (https://www.kernel.org/doc/Documentation/power/power_supply_class.txt).

BAT0

systemd Services

Resources utilization of systemd services. netdata monitors all systemd services via CGROUPS (the resources accounting used by containers).

cpu

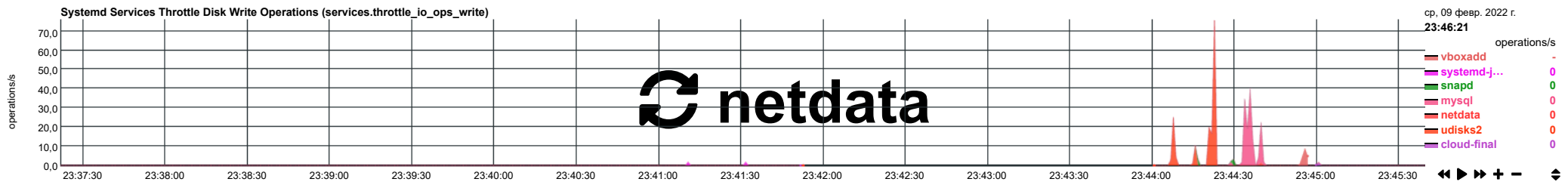
mem

disk

netdata

netdata

netdata



netdata

Applications

Per application statistics are collected using netdata's `apps.plugin` . This plugin walks through all processes and aggregates statistics for applications of interest, defined in `/etc/netdata/apps_groups.conf` , which can be edited by running `$ /etc/netdata/edit-config apps_groups.conf` (the default is here (https://github.com/netdata/netdata/blob/master/collectors/apps.plugin/apps_groups.conf)). The plugin internally builds a process tree (much like `ps fax` does), and groups processes together (evaluating both child and parent processes) so that the result is always a chart with a predefined set of dimensions (of course, only application groups found running are reported). The reported values are compatible with `top` , although the netdata plugin counts also the resources of exited children (unlike `top` which shows only the resources of the currently running processes). So for processes like shell scripts, the reported values include the resources used by the commands these scripts run within each timeframe.

cpu

disk

netdata

netdata

netdata

mem

Real memory (RAM) used by applications. This does not include shared memory.

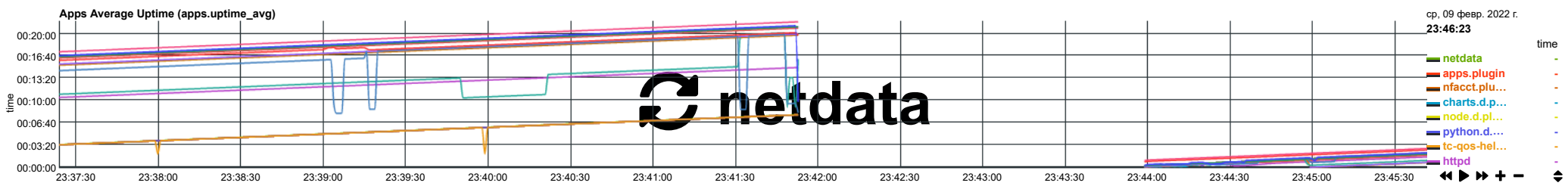
Virtual memory allocated by applications. Please check this article (https://github.com/netdata/netdata/tree/master/daemon#virtual-memory) for more information.

netdata

netdata

processes

Carried over process group uptime since the Netdata restart. The period of time within which at least one process in the group was running.



swap

net

User Groups

Per user group statistics are collected using netdata's `apps.plugin` . This plugin walks through all processes and aggregates statistics per user group. The reported values are compatible with `top` , although the netdata plugin counts also the resources of exited children (unlike `top` which shows only the resources of the currently running processes). So for processes like shell scripts, the reported values include the resources used by the commands these scripts run within each timeframe.

cpu

disk

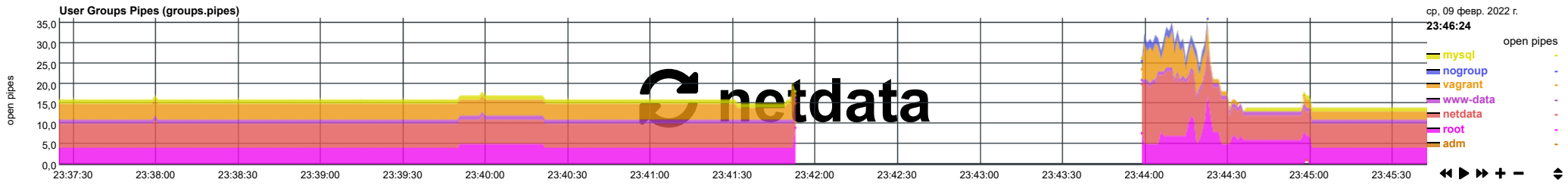
mem

Real memory (RAM) used per user group. This does not include shared memory.

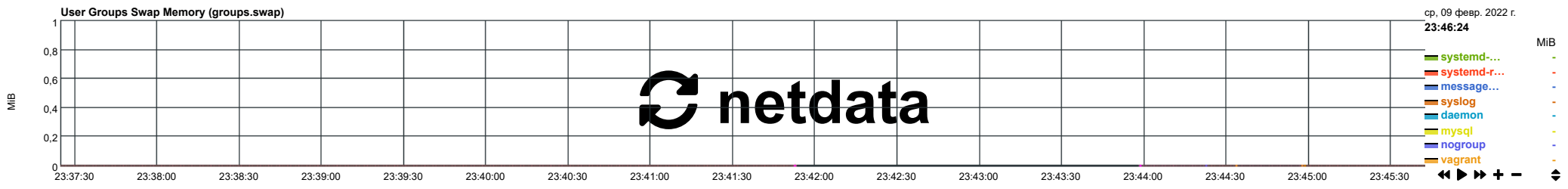
Virtual memory allocated per user group since the Netdata restart. Please check this article (https://github.com/netdata/netdata/tree/master/daemon#virtual-memory) for more information.

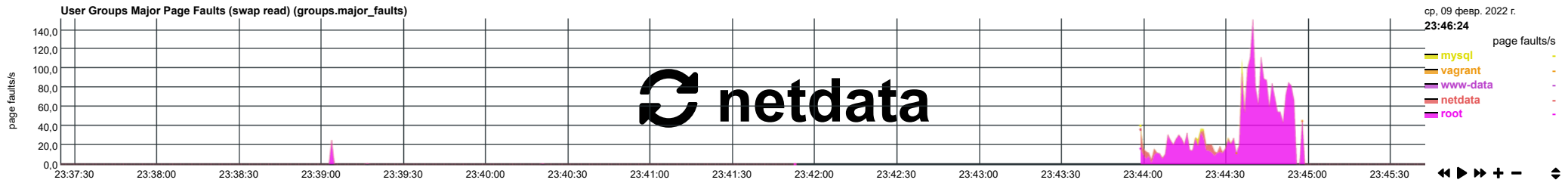
processes

Carried over process group uptime. The period of time within which at least one process in the group was running.

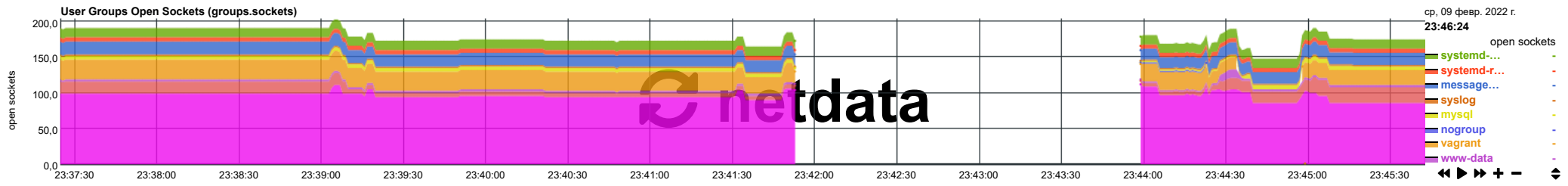


swap





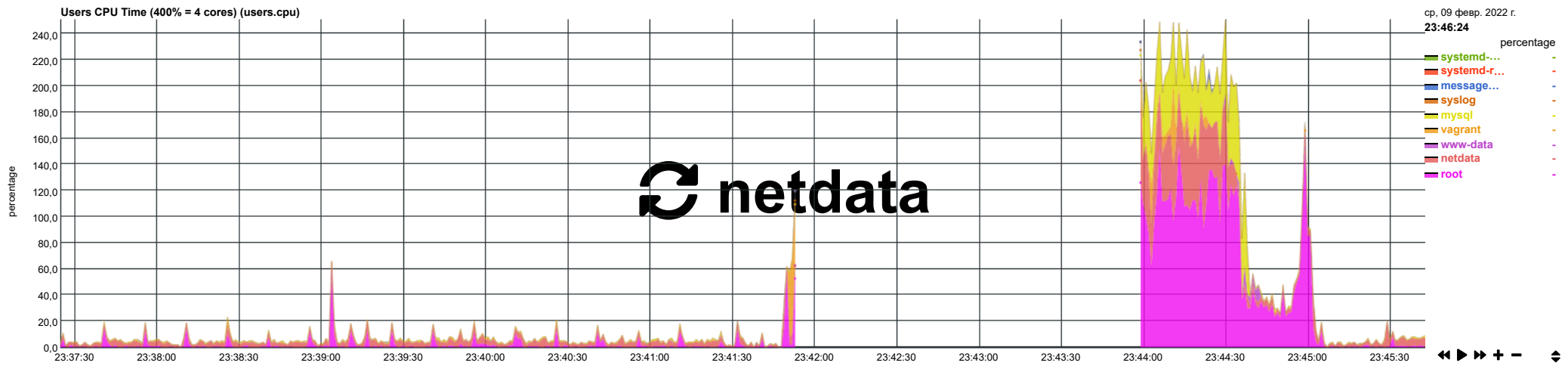
net



Users

Per user statistics are collected using netdata's `apps.plugin` . This plugin walks through all processes and aggregates statistics per user. The reported values are compatible with `top` , although the netdata plugin counts also the resources of exited children (unlike `top` which shows only the resources of the currently running processes). So for processes like shell scripts, the reported values include the resources used by the commands these scripts run within each timeframe.

cpu



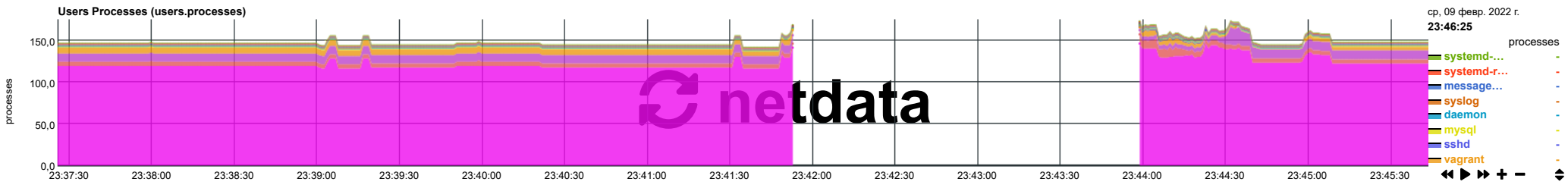
disk

mem

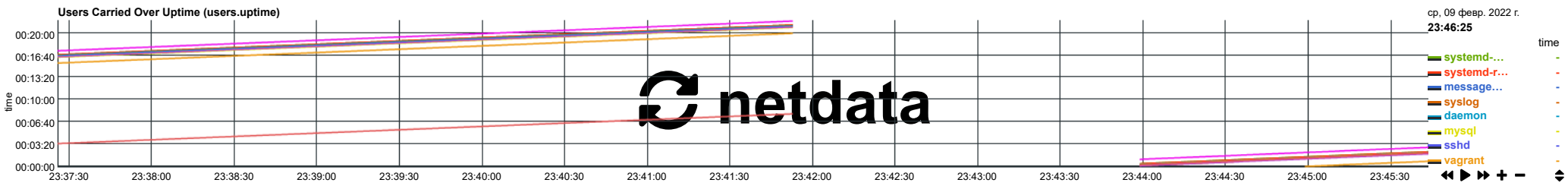
Real memory (RAM) used per user. This does not include shared memory.

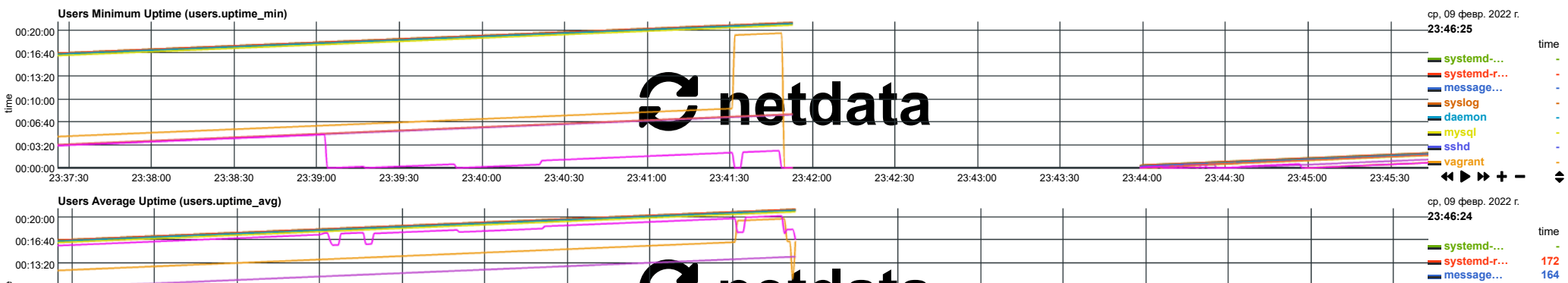
Virtual memory allocated per user. Please check this article (https://github.com/netdata/netdata/tree/master/daemon#virtual-memory) for more information.

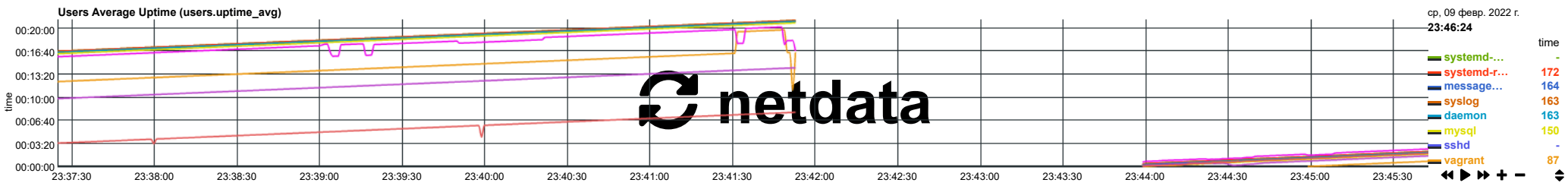
processes

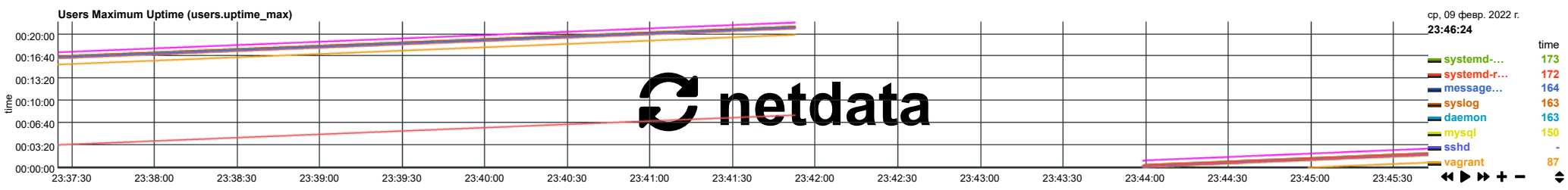


Carried over process group uptime since the Netdata restart. The period of time within which at least one process in the group was running.









swap

net

Apache local

requests

bandwidth

workers

statistics

Sensors

Readings of the configured system sensors.

voltage

web log apache vhosts

Information extracted from a server log file. `web_log` plugin incrementally parses the server log file to provide, in real-time, a break down of key server performance metrics. For web servers, an extended log file format may optionally be used (for `nginx` and `apache`) offering timing information and bandwidth for both requests and responses. `web_log` plugin may also be configured to provide a break down of requests per URL pattern (check `/etc/netdata/python.d/web_log.conf` (https://github.com/netdata/netdata/blob/master/collectors/python.d.plugin/web_log/web_log.conf)).

responses

Web server responses by type. `success` includes **1xx**, **2xx**, **304** and **401**, `error` includes **5xx**, `redirect` includes **3xx** except **304**, `bad` includes **4xx** except **401**, `other` are all the other responses.

Web server responses by code family. According to the standards `1xx` are informational responses, `2xx` are successful responses, `3xx` are redirects (although they include **304** which is used as "**not modified**"), `4xx` are bad requests, `5xx` are internal server errors, `other` are non-standard responses, `unmatched` counts the lines in the log file that are not matched by the plugin (let us know (https://github.com/netdata/netdata/issues/new?title=web_log%20reports%20unmatched%20lines&body=web_log%20plugin%20reports%20unmatched%20lines.%0A%0AThis%20is%20my%20log:%0A%0A%60%60%60txt%0A%0Aplease%20paste%20your%20web%20server%20log%20he
if you have any unmatched).

Number of responses for each response code individually.

bandwidth

Bandwidth of requests (`received`) and responses (`sent`). `received` requires an extended log format (without it, the web server log does not have this information). This chart may present unusual spikes, since the bandwidth is accounted at the time the log line is saved by the web server, even if the time needed to serve it spans across a longer duration. We suggest to use QoS (e.g. FireQOS (http://firehol.org/#fireqos)) for accurate accounting of the web server bandwidth.

http methods

http versions

ip protocols

Web server requests received per IP protocol version.

clients

Charts showing the number of unique client IPs, accessing the web server.

Unique client IPs accessing the web server, within each data collection iteration. If data collection is **per second**, this chart shows **unique client IPs per second**.

Unique client IPs accessing the web server since the last restart of netdata. This plugin keeps in memory all the unique IPs that have accessed the web server. On very busy web servers (several millions of unique IPs) you may want to disable this chart (check `/etc/netdata/python.d/web_log.conf` (https://github.com/netdata/netdata/blob/master/collectors/python.d.plugin/web_log/web_log.conf)).

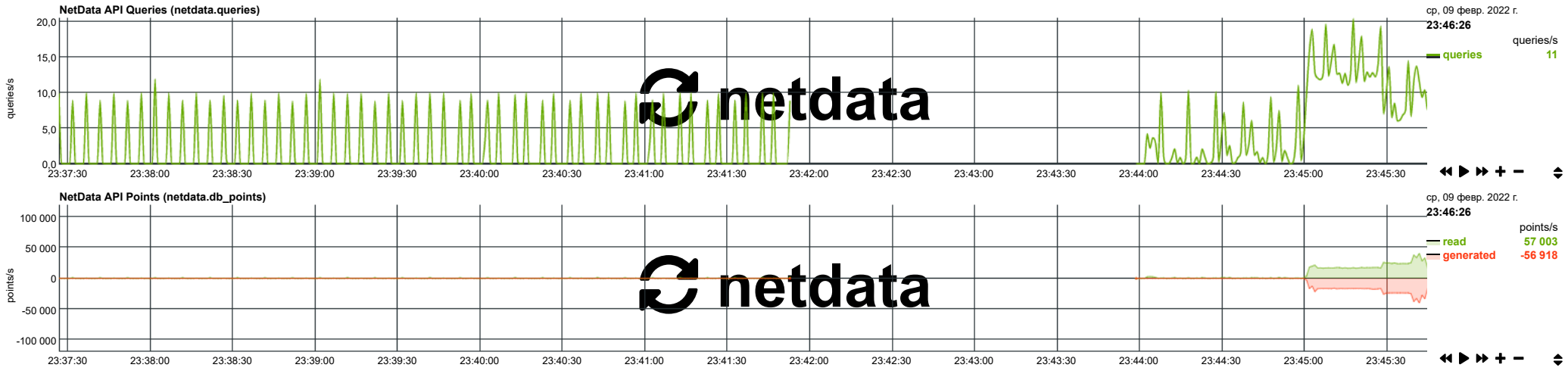
Netdata Monitoring

Performance metrics for the operation of netdata itself and its plugins.

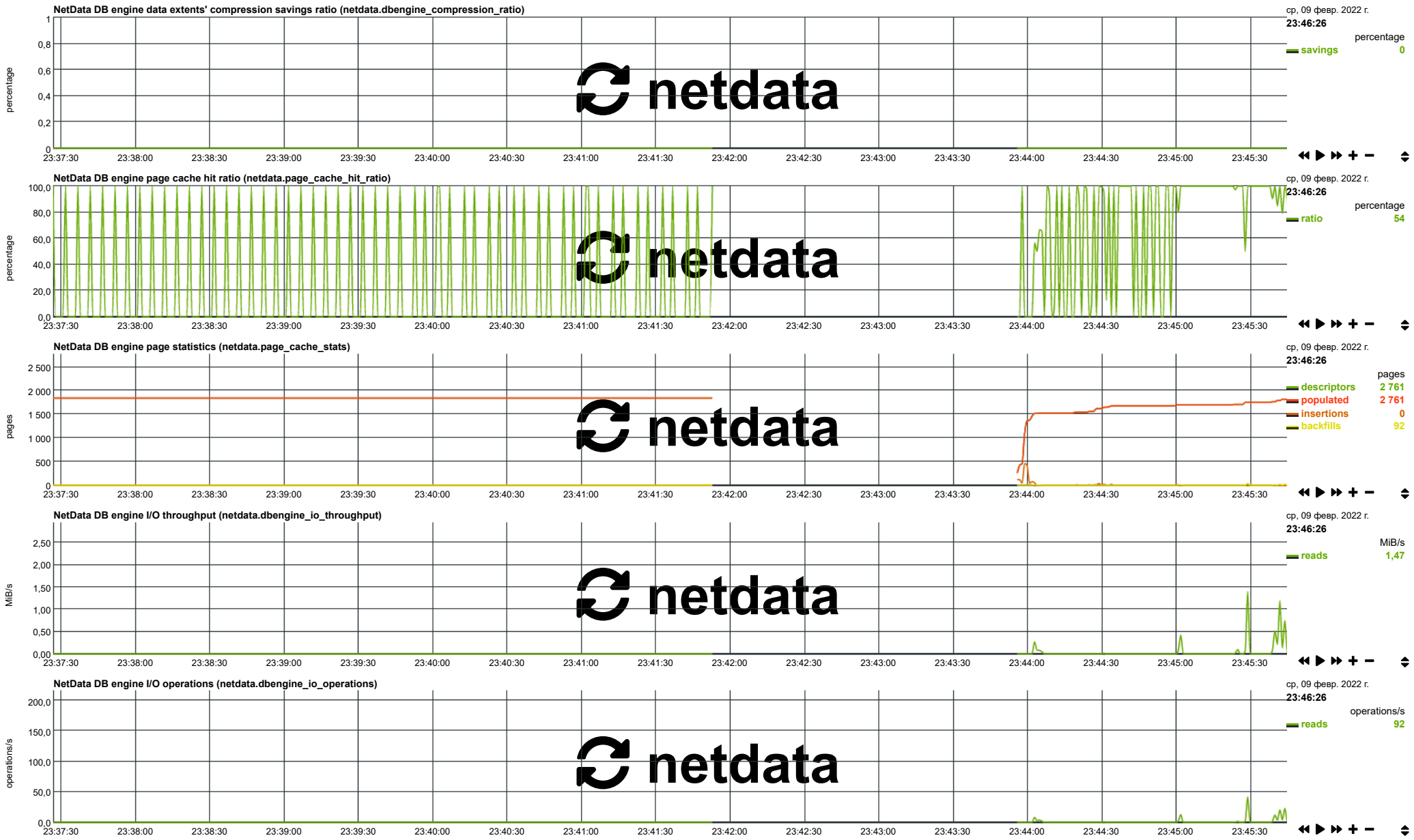
netdata

The netdata API response time measures the time netdata needed to serve requests. This time includes everything, from the reception of the first byte of a request, to the dispatch of the last byte of its reply, therefore it includes all network latencies involved (i.e. a client over a slow network will influence these metrics).

queries



dbengine



cgroups

proc

web


netdata


netdata

statsd


netdata


netdata


netdata


netdata


netdata


netdata


netdata

diskspace

tc.helper

apps.plugin

netdata

netdata

python.d

netdata

netdata

netdata

Netdata (https://github.com/netdata/netdata/wiki)

Copyright 2018, Netdata, Inc (mailto:info@netdata.cloud).
Copyright 2016-2018, Costa Tsaousis (mailto:costa@tsaousis.gr).

Released under GPL v3 or later (http://www.gnu.org/licenses/gpl-3.0.en.html). Netdata uses third party tools (https://github.com/netdata/netdata/blob/master/REDISTRIBUTED.md).